软件工程第一次大作业----mini-chatroom

18软件学院：孙一佳

# **1项目分解**

我们的目的是：实现一个局域网内的多人mini聊天室。最终提交的产品：可运行的局域网内的多人聊天室平台。

首先将整个项目通过WBS模型进行详细分解，如图1.1所示。

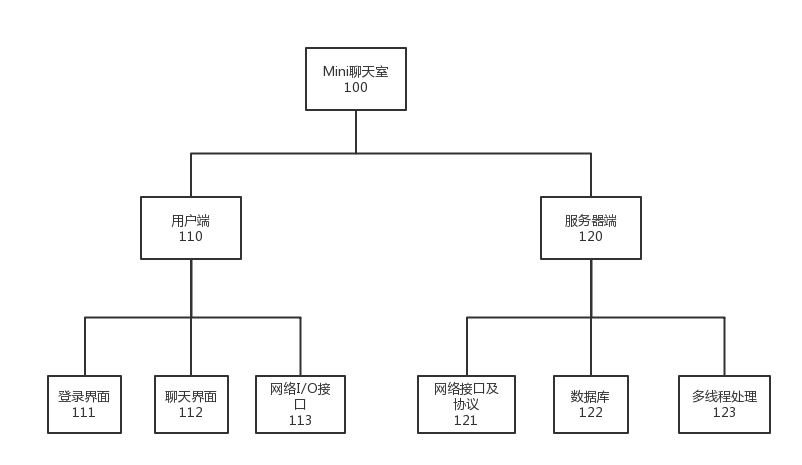


图1.1 WBS图

同时将图上的每一项进行细致的表述，详见下表1.1。

表1.1 项目工作分解结构表

|  |  |  |  |
| --- | --- | --- | --- |
| 项目名称：Mini聊天室 | | 项目负责人：孙一佳 | |
| 单位名称：中科大软件学院 | | 制表日期：2018.10.9 | |
| 工作分解结构 | | | |
| 任务编码 | 任务名称 | 主要活动表述 | 负责人 |
| 100 | Mini聊天室 | 项目名称 | 孙一佳 |
| 110 | 用户端 | 用户使用的终端 | 孙一佳 |
| 120 | 服务器端 | 数据存储与处理端 | 孙一佳 |
| 111 | 登录界面 | 用户登录 | 孙一佳 |
| 112 | 聊天界面 | 用户聊天 | 孙一佳 |
| 113 | 网络I/O接口 | 用户输入消息与接受处理后消息的接口 | 孙一佳 |
| 121 | 网络接口及协议 | 服务器端的消息接口，与消息传输协议 | 孙一佳 |
| 122 | 数据库 | 存储数据的载体 | 孙一佳 |
| 123 | 多线程处理 | 消息处理机制 | 孙一佳 |

# **2规模估算**

项目启动之前，进行了可行性研究与需求分析，同时将项目分解成每一个小的功能项，这有助于前期对项目规模的估算。同时项目估算的准确性也决定了我们对进度安排的合理性，风险管理的可靠性。

由于LOC估算法依赖于编程语言，并且对形势的变化不具有良好的弹性，下面将以FP估算法，对我们的mini聊天室进行规模的估算。

首先我们要定义数据项内容和结构，详见下表2.1.

表2.1 数据项

|  |  |  |
| --- | --- | --- |
| 用户 | 用户ID | 系统分配 |
| 用户名 | 用户可更改 |
| 用户IP地址 | 网络分配 |
| 消息 | 消息ID | 系统分配 |
| 消息的用户ID | 系统匹配 |
| 消息发送时间 | 系统分配 |
| 消息的内容 | 用户可更改 |

由上面的数据项分别对ILF，EIF，EI，EQ，EO的FP个数估计，详见下表2.2，2.3，2.4，2.5。其中**EIF外部接口文件无，因为只有一个系统。**

表2.2 ILF

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ILF内部逻辑文件** | **RET记录项类型个数** | **DET数据项个数** | **复杂度** | **未调整的FP个数** |
| 用户 | 用户基本信息 | 3 | 低 | 7 |
| 消息 | 消息基本信息 | 4 | 低 | 7 |

表2.3 EI

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **EI外部输入** | **FTR应用文件类型** | **DET个数** | **复杂度** | **未调整的FP个数** |
| 添加用户 | 用户 | 用户名，用户无权操作，1个 | 低 | 3 |
| 修改用户 | 用户 | 用户名，1个 | 低 | 3 |
| 添加消息（发送） | 消息 | 消息内容，其他都是系统分配，用户无权输入，1个 | 低 | 3 |

表2.4 EQ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **EQ外部查询** | **FTR应用文件类型** | **DET个数** | **复杂度** | **未调整的FP个数** |
| 查询用户 | 用户 | 用户ID，用户名，IP地址，共3个 | 低 | 3 |
| 查询消息（显示） | 消息 | 消息ID，消息用户ID，消息发送时间，消息内容，共4个 | 低 | 3 |

表2.5 EO

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **EO外部输出** | **FTR应用文件类型** | **DET个数** | **复杂度** | **未调整的FP个数** |
| 整理消息（服务器输出） | 消息 | 消息ID，消息用户ID，消息发送时间，消息内容，共4个 | 低 | 4 |

接下来通过系统特性调整，对FP进行调节，如下表2.6.

表2.6 系统特性调整

|  |  |  |
| --- | --- | --- |
| **序号** | **模块名称** | **最终结果** |
|  | **系统需要可靠的备份和恢复吗？** | 2 |
|  | **需要数据通信吗？** | 5 |
|  | **有分布处理的功能吗？** | 0 |
|  | **性能是否关键？** | 2 |
|  | **系统是否在一个已有的、使用的操作系统环境中运行？** | 3 |
|  | **系统需要联机数据项吗？** | 0 |
|  | **联机数据项是否需要在多屏幕或多操作之间切换已完成输入？** | 0 |
|  | **需要联机更新主文件吗？** | 0 |
|  | **输入、输出、文件或查询很复杂吗？** | 0 |
|  | **内部处理复杂吗？** | 0 |
|  | **代码需要设计成可复用吗？** | 0 |
|  | **设计中需要包括转换及安装吗？** | 0 |
|  | **系统的设计支持不同组织的多次安装吗？** | 1 |
|  | **应用的设计方便用户修改和使用吗？** | 5 |
| **合计：** |  | 18 |

综上，我们可以得到我们的FP估算的最终结果，由

可知：

# **3进度管理计划**

为确保项目如期完成，预先制定好进度管理计划是必不可少的。在进度管理计划中，合理安排好每个任务的时间跨度，既可以如期完成任务，也能高效地利用好时间。接下来通过甘特图和里程碑图来详细展示每一个环节的进度管理计划。甘特图如图3.1，里程碑图3.2.

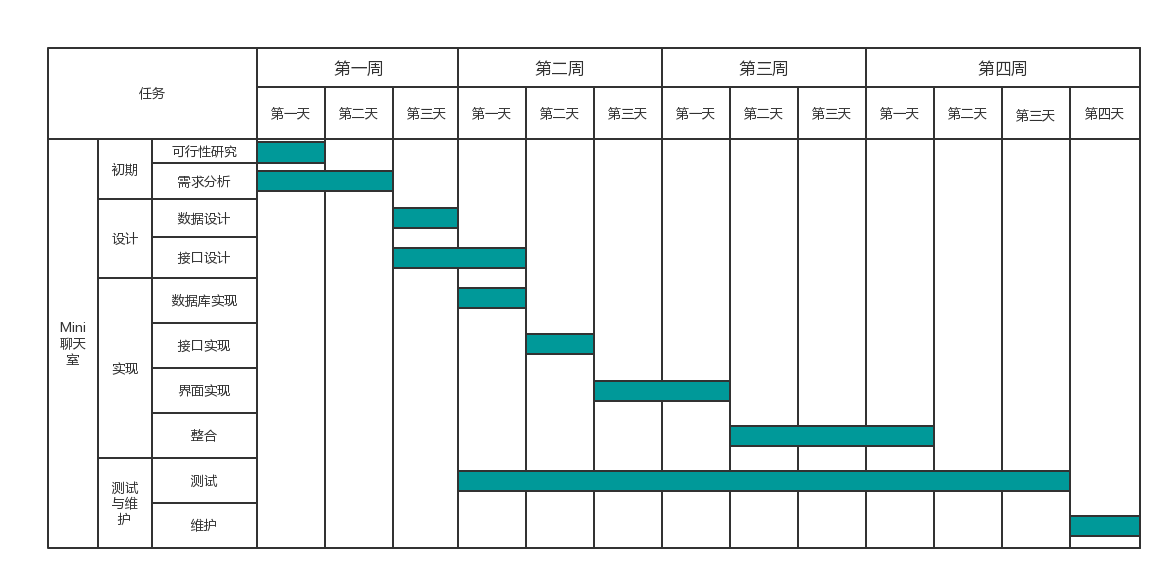


图3.1 甘特图

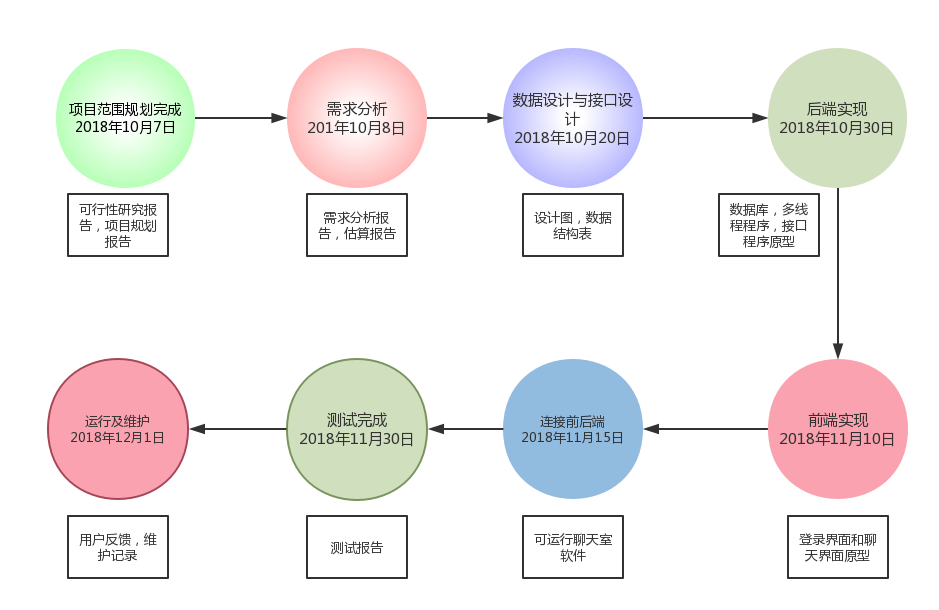


图3.2 里程碑图

其中我们在甘特图中将任务分为4周进行，希望能在四周内完成项目，其中我们余下了1周，可以作为弹性的管理，在某个环节出现延期时，可以及时弥补上时间的缺陷，不出意外，4周的时间足以解决项目。

# **4风险管理**

风险是现实情况与期望之间的差距，由于人力、技术、进度等等的不可预测因素，项目是可能发生变化的，从而导致项目无法完成或者延期完成等风险。因此我们应该在早期就将风险标识出来，同时预测以及评估它们，最后找出原因，管理和规避好风险的发生。

首先我们要从多个角度去标识风险，接下来列举人力、技术以及进度三个方向的风险标识。详见下表4.1，4.2，4.3.

表4.1 人员风险表

|  |  |  |  |
| --- | --- | --- | --- |
| **问 题** | **风险程度（0~5）** | **发生概率（0~1）** | **后果及影响** |
| 开发人员的水平如何； | 1 | 0.1 | 学习的时间过长，导致项目无法如期完成 |
| 开发人员在技术上是否配套； | 2 | 0.3 | 需要更换技术，将会把时间延长，无法如期交付 |
| 开发人员是否能自始至终地参加软件开发工作； | 0 | 0.1 | 一个人的工程实验，人在项目在。 |
| 开发人员能否集中精力开发； | 1 | 0.2 | 精力不集中导致效率低下，错误频出。 |
| 开发人员是否对工作有正确的期望； | 2 | 0.2 | 实现了错误的需求，功能不符合既定要求。 |
| 开发人员是否受到过培训； | 3 | 0.1 | 未受过培训，自己需要查找资料自学，时间延长。 |

表4.2 技术风险表

|  |  |  |  |
| --- | --- | --- | --- |
| **问 题** | **风险程度（0~5）** | **发生概率（0~1）** | **后果及影响** |
| 开发人员的工程技术水平如何； | 3 | 0.4 | 功能实现不全，延期交付。 |
| 开发人员的编程技术是否合格； | 1 | 0.1 | 无法如期交付。 |
| 实现项目的技术是否成熟； | 0 | 0.1 | 技术的本身存在问题，运行时会产生无法预知的问题。 |
| 有可靠的技术资源文档吗； | 1 | 0.1 | 没有参考会导致错误的理解和错误的实现。 |
| 导师是否有技术指导； | 2 | 0.1 | 遇到问题无法解决，项目无法正常推进。 |

表4.3 进度风险表

|  |  |  |  |
| --- | --- | --- | --- |
| **问 题** | **风险程度（0~5）** | **发生概率（0~1）** | **后果及影响** |
| 是否合理地安排进度； | 3 | 0.3 | 无法按时完成任务，延迟交付。 |
| 项目的期限是否足够开发； | 2 | 0.1 | 无法如期完成功能和软件。 |
| 开发人员是否能从始至终地开发； | 0 | 0 | 人在项目在，人走项目废。 |
| 是否有合理的进度督促机制； | 2 | 0.6 | 进度推进慢，延期交付。 |
| 在每个里程碑时刻是否有完整的测试机制； | 4 | 0.5 | 软件在后期出问题，并且难以锁定问题位置。 |

接下来评估其中较为严重的风险：（超过30%且风险程度高于2的）：详见表4.4

表4.4 风险评估

|  |  |  |  |
| --- | --- | --- | --- |
| **风险** | **风险概率** | **风险影响** | **风险参考水准** |
| 开发人员在技术上是否配套 | 0.3 | 2 | 前期阅读资料，并做出合理报告，第三周时需要有网络原型，第四周需要有界面原型，第六周要有整合好的第一次运行原型。 |
| 开发人员的工程技术水平如何 | 0.4 | 3 | 在甘特图的时间安排下，完成进度的水平对比，如果落后过多，需要增加每周的学习与工作时间。 |
| 是否合理地安排进度 | 0.3 | 3 | 根据甘特图和WBS图参考。 |
| 是否有合理的进度督促机制 | 0.6 | 2 | 根据老师和助教师兄的作业安排作为督促标准。 |
| 在每个里程碑时刻是否有完整的测试机制 | 0.5 | 4 | 在里程碑时刻，与同项目同学互相测试，并给出优缺点提示。 |

由上述的分析可以找出风险的潜在原因，并想办法管理与监控风险。我们有两种方法去解决，第一种是风险避免，主动地避免风险，分析风险的原因，采取措施。

我们安排如下：对于人员和进度上的风险，最大的原因在于负责人的学习与精力的持续性。合理安排时间，制定长短期计划表，暂定50天实现，共分成7周，前5周每周3天抽出一个下午，刚好是4个小时，认真地学习并完成一个功能模块，最后两周视情况增减时间。

第二种是风险监控，监控可以提供风险指示的因素。

对于技术上的风险，设定评估模块的进度和时间安排，具体见进度管理，在这个约束下，根据时间的推进，随时监督自己的技术完成度，及时地补充知识。

# **5软件质量管理**

想要得到一个高效且优秀的系统和项目，仅仅上述的环节还不够，我们需要提前对它的质量进行管理，定下标准。在此项目中，我们通过质量控制的方法，保证工作产品满足需求并且应用于整个开发周期中的一系列审查、评审和测试。

质量控制在创建工作产品的过程中包含一个反馈循环，并定义好可度量的规约，并将每个过程的产品与这一规约进行比较。

（1）技术手段的规约：C++编程语言的熟悉和运用程度（0-5），socket编程的熟悉和运用程度（0-5），前端界面制作的学习程度（0-5），在每个阶段进行评估，0为完全不熟悉，5为熟练运用，要求每一周和值递增2-3，如果低于此标准，要及时增加时间弥补。

（2）加强软件测试：对每一个里程碑时刻产生的产品和每一个功能模块产生的原型，要进行可度量的规约测试，测试标准参考甘特图以及老师提供的标准。

（3）软件的修改、变更：此次软件的管理交由github管理，每周有三天进行工作和变更文档，每天两次同步分布库，根据次数来控制更改。

（4）软件质量：重要功能原型制定质量测试计划，socket编程部分及其接口质量（0-5），界面功能及美观度（0-5），数据库结构与效率（0-5），与同项目同学互相打分，随时控制质量。